PCT/EP2004/051233

08. 07. 2004

INVESTOR IN PEOPLE

The Patent Office
Concept House
Cardiff Road
Newport
South Wales
NP10 8QQ

RECD 27 JUL 2004
WIPO PCT

PRIORITY DOCUMENT

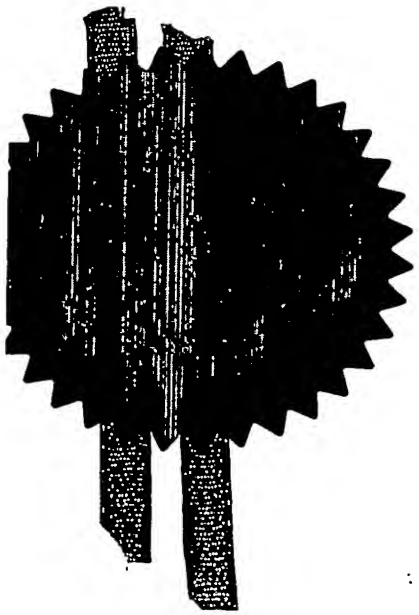
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

I, the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration save for the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so re-registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P.L.C. or PLC.

Re-registration under the Companies Act does not constitute a new legal entity but merely subjects the company to certain additional company law rules.



Signed

Dated 29 June 2004

CERTIFIED COPY OF PRIORITY DOCUMENT

Patents Form 1/77

The Patent Office

Patents Act 1977
(Rule 16)

THE PATENT OFFICE
DL
27 JUN 2003
RECEIVED BY FAX

27JUN03 E816244-1 D01065
P01/7700 0.00-0314971.3

Request for grant of a patent

(See the notes on the back of this form. You can also get an explanatory leaflet from the Patent Office to help you fill in this form)

27 JUN 2003

The Patent Office
Cardiff Road
Newport
South Wales
NP9 1RH

1. Your reference — DTP.P52786GB

2. Patent application number *(The Patent Office will fill in this part)* — 0314971.3

3. Full name, address and postcode of the or of each applicant *(underline all surnames)* — Telefonaktiebolaget LM Ericsson (PUBL)
SE-12625
Stockholm
Sweden

Patents ADP number *(if you know it)* — 08455388001

If the applicant is a corporate body, give the country/state of its incorporation — Sweden

4. Title of the invention — Method for Distributing Passwords

5. Name of your agent *(if you have one)* — Marks & Clerk

"Address for service" in the United Kingdom to which all correspondence should be sent *(including the postcode)* — 4220 Nash Court
Oxford Business Park South
Oxford OX4 2RU
United Kingdom

Patents ADP number *(if you know it)* — 7271125001

6. If you are declaring priority from one or more earlier patent applications, give the country and the date of filing of the or of each of these earlier applications and *(if you know it)* the or each application number

Country	Priority application number *(if you know it)*	Date of filing *(day / month / year)*

7. If this application is divided or otherwise derived from an earlier UK application, give the number and the filing date of the earlier application

Number of earlier application	Date of filing *(day / month / year)*

8. Is a statement of inventorship and of right to grant of a patent required in support of this request? *(Answer 'Yes' if:*
a) any applicant named in part 3 is not an inventor, or
b) there is an inventor who is not named as an applicant, or
c) any named applicant is a corporate body.
See note (d)) — Yes

Patents Form 1/77

Patents Form 1/77

Enter the number of sheets for any of the following items you are filing with this form. Do not count copies of the same document

Continuation sheets of this form	
Description	9
Claim (s)	1
Abstract	1
Drawing (s)	

10. If you are also filing any of the following, state how many against each item.

Priority documents	
Translations of priority documents	
Statement of inventorship and right to grant of a patent *(Patents Form 7/77)*	
Request for preliminary examination and search *(Patents Form 9/77)*	
Request for substantive examination *(Patents Form 10/77)*	
Any other documents *(please specify)*	

11. I/We request the grant of a patent on the basis of this application.

Signature: Marks & Clerk — Date: 27 June 2003

12. Name and daytime telephone number of person to contact in the United Kingdom: Dr. Daniel Talbot-Ponsonby 01865-397900

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least 6 weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

a) *If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500505.*

b) *Write your answers in capital letters using black ink or you may type them.*

c) *If there is not enough space for all the relevant details on any part of this form, please continue on a separate sheet of paper and write "see continuation sheet" in the relevant part(s). Any continuation sheet should be attached to this form.*

d) *If you have answered 'Yes' Patents Form 7/77 will need to be filed.*

e) *Once you have filled in the form you must remember to sign and date it.*

f) *For details of the fee and ways to pay please contact the Patent Office.*

Method for Distributing Passwords

Creation and distribution of end-user passwords for applications and application proxies is problematic. On the one hand, end-users tend to create passwords that may be simple, short and easily figured out by unauthorized parties. On the other hand, finding out the relationship between the end-user identities and their passwords is often insecure and costly.

Even though there are mechanisms that allow the re-use of already existing passwords between different trust domains, such as Single-Sign-On solutions developed by Liberty Alliance, it is often required to relay on a third party to perform the authentication. In this kind of model, the service provider does not actively take part in the authentication. On the other hand, the password management performed by the service provider is not always appropriate because the handling of passwords in application servers requires complex and costly management procedures, e.g. maintenance of the password lifetimes, overlapping password values, policies regarding valid values, and so on. Also, the end-users do not want to remember many passwords, and they tend to use the same passwords with several application servers, which clearly decrease the level of security.

HTTP Digest authentication framework (RFC 2617 and potential extensions) may include methods that are able to generate end-user passwords. For example, HTTP Digest AKA (RFC 3310 and its potential extensions) is a method for creating end-user passwords using existing 3GPP authentication infrastructure based on AKA credentials stored on tamper-resistant device, the so-called ISIM/USIM/SIM card. Details can be found at http://www.ietf.org/rfc.html Even though HTTP Digest provides a flexible way to generate fresh passwords without user involvement, it does not include a standard way to further delegate these passwords to third parties, such as application servers or proxies in visited networks. Furthermore, the current standards typically assume that the passwords generated using HTTP Digest can only be used once. This invention describes how the passwords generated using HTTP Digest AKA can be

delegated to third parties so that the passwords can be securely used for subsequent authentication.

The initial authentication is done between the end-user device and its home network using HTTP Digest with algorithm that is able to generate passwords, for example using HTTP Digest AKA (RFC 3310 and its potential extensions). During the initial authentication, the home network initiates a process in which the new password generated during HTTP Digest AKA is linked to the identity of a third party, such as an application server or a proxy in the visited network, and to a new temporary end-user identity for that third party. The end-user device and the third party can start using the new password and related identities using HTTP Digest (RFC 2617 and its potential extensions) as authentication method.

The solution works in the following way if HTTP Digest AKA is used:

1) the end-user sends a HTTP request to an application server (the third party) that does not share password with the end-user;
2) the challenge is sent to an authentication node in the home network (different procedures will apply); the identity of the application server may be included in the message;
3) the authenticator authenticates (or helps the application server to authenticate) the end-user in the way that the new HTTP Digest AKA password generated during the authentication can be used with the application server (third party); the identity of the application server (e.g. the "realm") and the identity of the end-user in that application server (e.g. the "username") are told to the end-user device by including this information to the HTTP Digest AKA authentication challenge;
4) the application server (third party) and the end-user device can start using the new password and identities when authenticating each other using HTTP Digest (RFC 2617). Depending on the implementation, the password can be given to the application server (third party), or the authenticator can keep the password,

and perform the authentication without revealing the password to the application server (third party).

HTTP Digest authentication framework use the following concepts that are also used in this document:

- 'realm': A string by which the client side knows which username and password to use. This string should contain at least the name of the host performing the authentication and might additionally indicate the collection of users who might have access. An example might be registered_users@home.com". In 3GPP/AKA context, the realm of the home network is typically stored in SIM/USIM/ISIM card.

- 'username': The user's name in the specified realm. This string is used in the server side the find the correct password for the user. In 3GPP/AKA context, the username used with the home network is typically stored in SIM/USIM/ISIM card. In most cases, the username is the same as the so-called Private Identity (IMPI). 3GPP username identifies the subscription, and for this reason the passwords are specific to the end-user device rather than to the real end-user. Note also that in normal HTTP authentication framework, the username and password are typed in by the end-user, but in 3GPP/AKA context these fields are automatically filled by end-user device.

The system has three entities: 1) the end-user device (UE) with HTTP Digest AKA implementation, 2) application server (AS), and 3) authenticator.

Step 1a

In the first phase (see figure 1a), the UE is initially authenticated using HTTP Digest AKA, and the created password is tied to the AS, and UE identities.

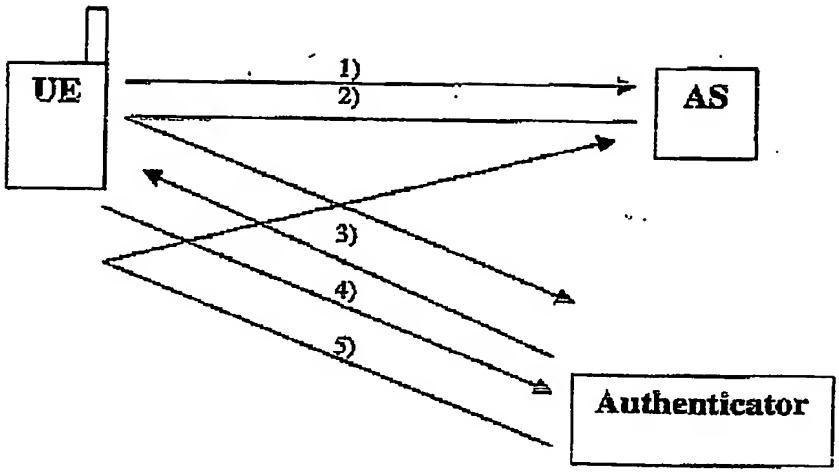


Figure 1a: Password generation and identity agreement

The procedure has the following steps:

1) UE sends a HTTP request (typically HTTP GET) to an AS.

2) Since the AS and the UE do not have a shared secret, the AS redirects the request to the Authenticator. Before redirecting the request, the AS may define a new temporary username for the end-user. The AS includes this username and its own identity information to the request (typically the identity information is encoded into the URI parameter in some standard format, e.g. "username@realm"). How the AS knows the identity of the Authenticator is out of the scope of this innovation. For example, AS may use the 'realm' parameter in the HTTP Digest AKA Authorization header to locate the authenticator.

3) The Authenticator checks if the AS is authorized to do HTTP re-direction and request a new HTTP Digest AKA password for this UE. If yes, then the Authenticator takes the identity information from the request (typically for the URI parameter), and encodes this information again to the HTTP Digest AKA challenge (most likely the Authenticator puts the identity information to the so-called "server data" of HTTP Digest AKA nonce, as described in RFC 3310).

Note: By including the identity to the challenge, the Authenticator can be sure that the identity of the AS or the temporary identity of the end-user cannot be changed by any party (such as an attacker) between the UE and Authenticator because the challenge is returned back to the Authenticator in the next message.

4) The UE authenticates the network (as defined in standard AKA protocol) and generates a new password based on HTTP Digest AKA challenge. The UE stores locally the identity of the AS and the newly generated password to be used later for mutual authentication with the AS. If the challenge included also a new temporary 'username' to the AS, the username is also stored with the password and the AS identity. Both the AS and UE identities are encoded in the HTTP Digest AKA challenge, .e.g using the format "username@realm". UE sends the authentication response to the Authenticator.

Note: The UE should mark the potential new 'username' as temporary username. This username and related HTTP Digest AKA password should be removed when new potential 'username' and password are generated for the same realm. If the challenge does not include new temporary username, then the existing username should be re-used.

5) The Authenticator authenticates the UE, and if successful, stores the new password and the identities of the UE and the AS to be used later. The request is redirected back to the AS.

What happens next is up to the implementation. If appropriate, the AS may trust on the initial authentication performed by the Authenticator. If not perceived secure, the AS may also re-challenge the UE now using the newly generated password. This time, HTTP Digest AKA is not used for authentication. Instead, HTTP Digest with some other algorithm is used, e.g. MD5 can be used (see RFC 2617).

Step 1b

The first phase can also be performed using a different procedure (see figure 1b).

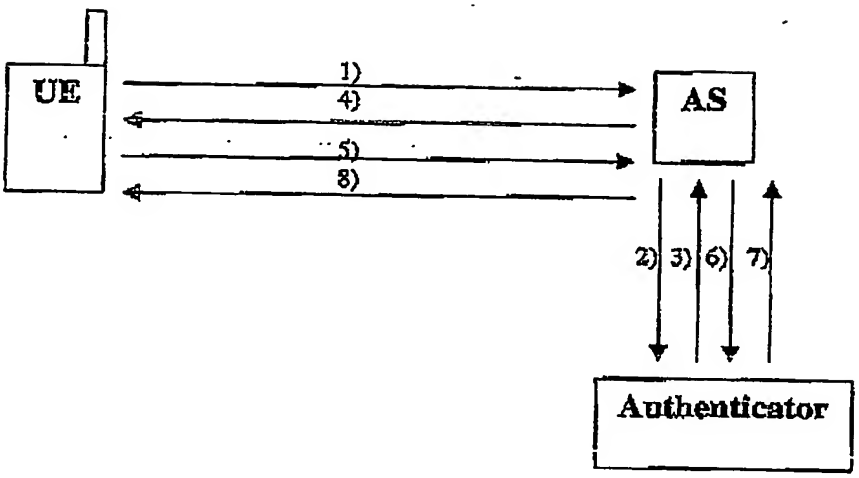


Figure 1b: Password generation and identity agreement

The procedure has the following steps:

1) UE sends a HTTP request (typically HTTP GET) to an AS.

2) Since the AS and the UE do not have a shared secret, the AS requests HTTP Digest AKA authentication challenge directly from the Authenticator. The request may include the identity of the AS and the new temporal identity of the end-user.

3) The Authenticator checks if the AS is authorized to request a new HTTP Digest AKA password for this UE. If yes, then the Authenticator takes the identity of the AS and the temporal identity of the end-user (if present) from the request, and encodes this information to the HTTP Digest AKA challenge as in Figure 1a. Alternatively, the AS may also include this information to the challenge in step 4. Identity information is encoded in some standard format, e.g. username@realm. Information needed by the AS to create HTTP Digest AKA challenge are sent back to the AS. If these parameters include the HTTP Digest AKA password, then process steps 6) and 7) may not be needed.

4) The UE is challenged by HTTP Digest AKA authentication challenge. The AS may add the identity information (both the AS and end-user identities) to the

authentication challenge before sending the challenge to the UE. However, in this case, the AS must be the end-point for the authentication, and possess the HTTP Digest AKA password.

5) The UE authenticates the network (as defined in standard AKA protocol) and generates a new password based on HTTP Digest AKA challenge. The UE stores locally the identity of the AS (e.g. the "realm"), the newly generated password, and the new temporary 'username' for itself to the AS to be used later for mutual authentication with the AS – if present in the challenge. UE sends the authentication response to the AS.

Note: The UE should mark the potential new 'username' as temporary username. This username and related HTTP Digest AKA password should be removed when new 'username' and password is generated for the same realm. If the challenge does not include new temporary username, then the existing username should be re-used.

6) If the AS did not receive the end-user password in step 3), it must request Authenticator to do the authentication at this phase.

7) If the AS did not receive the end-user password in step 3) and it requested authentication in step 6, the Authenticator authenticates the UE, and returns the appropriate result to the AS. The Authenticator may also send the end-user password to AS at this or some later phase.

8) If the UE authentication was successful, the service is delivered to the UE.

Step 2

When the AS wants next time to authenticate the UE (which may be directly after the previous procedures, or after some longer period of time, e.g. when the UE contacts the AS next time), the procedure is as presented in Figure 2.

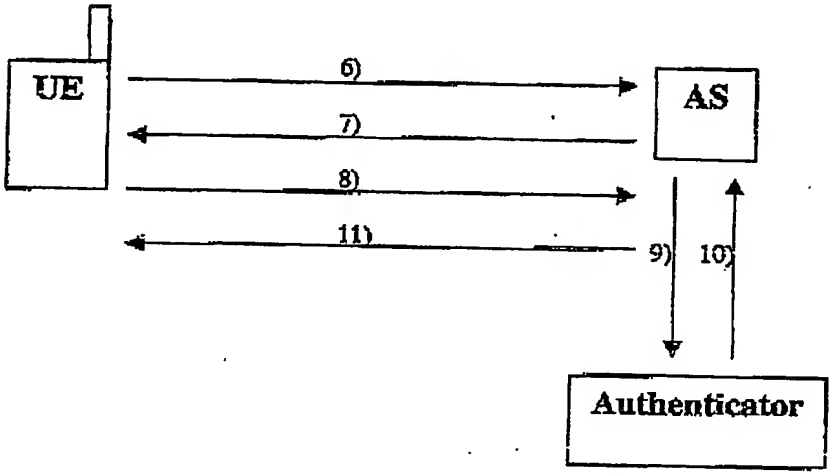


Figure 2: The use of new password.

6) UE sends a HTTP request (typically HTTP GET) to an AS.

7) Since the AS and the UE have now a shared secret, the AS challenges the UE with HTTP Digest challenge. The challenge includes the identity of the AS in the "realm" parameter.

8) The UE sends an authentication response (typically in HTTP GET request) back to the AS using the new temporary 'username' and password for the AS created during the previous phase. If the new temporary username was not created, then the normal AKA specific username is used. The UE uses the "realm" parameter to identify the correct password.

9) If the AS possesses the end-user password, steps 9 and 10 are not needed. If not, then the AS request from the Authenticator (or some other network entity to where the Authenticator has stored the UE specific password), for authentication.

10) There are two different modes to proceed:

a) The Authenticator may take care of the authentication on behalf of the AS. In this case, the AS does not need to know the password, and the

Authenticator returns just information on whether the authentication was successful or not.

b) The Authenticator may send the password to the AS, and the AS may perform the authentication.

11) If the authentication was successful, the AS delivers the service to the UE.

CLAIMS:

1. A method of generating a password for use by an end-user device (UE) to access a remote server, comprising:

sending a request for access from the UE to the remote server;

sending to an authentication node in the UE's home network details of the request for access and the identity of the remote server;

at the authentication node or the remote server, generating a HTTP Digest challenge using an algorithm capable of generating end-user passwords, such as HTTP Digest AKA, including details of the identity of the remote server and the identity of the UE;

at the UE, generating a password based on the HTTP Digest challenge, said password being associated with the identity of the remote server and the identity of the UE; and

storing the password at the UE.

ABSTRACT

Method for Distributing Passwords

A method of generating a password for use by an end-user device (UE) to access a remote server comprises sending a request for access from the UE to the remote server, and sending details of the request for access and the identity of the remote server to an authentication node in the UE's home network. The authentication node generates a HTTP Digest challenge with algorithm that is able to generate passwords for the UE, including details of the identity of the remote server and the identity of the UE in the challenge. The UE generates a password based on the HTTP Digest challenge. The password is associated with the identity of the remote server and the identity of the UE and is stored at the UE.